Activity recognition using the velocity histories of tracked keypoints

Ross Messing
University of Rochester
rmessing@cs.rochester.edu

Chris Pal
École Polytechnique de Montréal
christopher.pal@polymtl.ca

Henry Kautz
University of Rochester
kautz@cs.rochester.edu

Abstract

We present an activity recognition feature inspired by human psychophysical performance. This feature is based on the velocity history of tracked keypoints. We present a generative mixture model for video sequences using this feature, and show that it performs comparably to local spatio-temporal features on the KTH activity recognition dataset. In addition, we contribute a new activity recognition dataset, focusing on activities of daily living, with high resolution video sequences of complex actions. We demonstrate the superiority of our velocity history feature on high resolution video sequences of complicated activities. Further, we show how the velocity history feature can be extended, both with a more sophisticated latent velocity model, and by combining the velocity history feature with other useful information, like appearance, position, and high level semantic information. Our approach performs comparably to established and state of the art methods on the KTH dataset, and significantly outperforms all other methods on our challenging new dataset.

1. Introduction

Activity recognition is an important area of active computer vision research. Recent work has focused on bag-of-spatio-temporal-features approaches that have proven effective on established datasets. These features have very strong limits on the amount of space and time that they can describe. Human performance suggests that more global spatial and temporal information could be necessary and sufficient for activity recognition. Some recent work has attempted to get at this global information by modeling the relationships between local space-time features. We instead propose a feature, directly inspired by studies of human performance, with a much less limited spatio-temporal range.

Our work is motivated an important application of activity recognition, assisted cognition health monitoring systems designed to unobtrusively monitor patients. These systems are intended to ensure the mental and physical health of patients either at home or in an extended care facility. Additionally, they provide critical, otherwise unavailable information to concerned family members or physicians making diagnoses. With an aging population, and without a concurrent increase in healthcare workers, techniques that could both lessen the burden on caregivers and increase quality of life for patients by unobtrusive monitoring are very important. We introduce a high resolution dataset of complicated actions designed to reflect the challenges faced by assisted cognition activity recognition systems, as well as other systems dealing with activities of daily living.

2. Background

Initial work on activity recognition involved extracting a monolithic description of a video sequence. This could have been a table of motion magnitude, frequency, and position within a segmented figure [16] or a table of the presence or recency of motion at each location [6]. Both of these techniques were able to distinguish some range of activities, but because they were individually the full descriptions of a video sequence, rather than features extracted from a sequence, it was difficult to use them as the building blocks of a more complicated system.

Another approach to activity recognition is the use of explicit models of the activities to be recognized. Domains where this approach has been applied include face and facial expression recognition [4] and human pose modeling [17]. These techniques can be very effective, but by their nature, they cannot offer general models of the information in video in the way that less domain-specific features can.

Recent work in activity recognition has been largely based on local spatio-temporal features. Many of these features seem to be inspired by the success of statistical models of local features in object recognition. In both domains, features are first detected by some interest point detector running over all locations at multiple scales. Local maxima of the detector are taken to be the center of a local spatial or spatio-temporal patch, which is extracted and summarized by some descriptor. Most of the time, these features are then clustered and assigned to words in a codebook, allowing the use of bag-of-words models from statistical natural language processing.

While local spatio-temporal features were initially introduced in bag-of-features approaches ignoring non-local spatio-temporal relationships between the local features, more recent work has attempted to weaken these spatio-temporal independence assumptions [10, 12, 15, 19]. While these techniques weaken either or both of the spatial and temporal independence assumptions, they are fundamentally limited by the locality of the features underlying their models, and the kinds of complicated motion they can disambiguate. Junejo *et al*.'s temporal self-similarity features sacrifice all spatial information for extraordinary viewpoint invariance[10]. Laptev *et al*.'s system caputres spatio-temporal relationships between a set of very coarse spatial and temporal grid regions[12]. Niebles and Fei-Fei's system creates a spatial constellation model of feature location within each frame, but does not consider relationships across frames[15]. Savarese *et al*. add a new type of feature to their bag of words that captures the spatio-temporal correlation of different types of features[19]. This describes broad relationships of the over space and time, but fails to capture the motion of any individual feature, or even type of feature.

Local features have proven successful with existing activity recognition datasets. However, most datasets contain low resolution video of simple actions. Since the expressiveness of a local feature decreases with the volume of space and time it can describe, as high resolution video becomes more commonplace, local features must lose information by operating on scaled-down videos, become larger, dramatically expanding the amount of computation they require, or represent a smaller and smaller space-time volume as video resolution increases. Additionally, as actions become more and more complicated, local space-time patches may not be the most effective primatives to describe them.

In proposing a feature as an alternative to local patch methods, we see that human subjects can disambiguate actions using only the motion of tracked points on a body, in the absence of other information. Johansson [9] showed that point-light displays of even very complicated, structured motion were perceived easily by human subjects . Most famously, Johansson showed that human subjects could correctly perceive "point-light walkers", a motion stimulus generated by a person walking in the dark, with points of light attached to the walker's body. In computer vision, Madabhushi and Aggarwal were able to disambiguate activities using only the tracked motion of a single feature, the subject's head. While their system had limited success, it's ability to use an extremely sparse motion feature to recognize activities shows just how informative the motion of a tracked point can be. Inspired by these results, we propose a new feature, the velocity history of tracked keypoints.

Much of the previous trajectory tracking work has often been based on object centroid or bounding box trajectories. In a more sophisticated extension of such approaches, Rosales and Sclaroff [18] used an extended Kalman filter to help keep track of bounding boxes of segmented moving objects and then use motion history images and motion energy images as a way of summarizing the trajectories of tracked objects. Madabhushi and Aggarwal [14] used the trajectories of heads tracked in video and built simple but effective models based on the mean and covariance of velocities. Jiang and Martin [8] build a graph template of keypoint motion, and match it to regions. While this technique includes keypoints with potentially significant spatial and temporal extent, it lacks a probabilistic formulation, and is expensive to run exactly. Our approach to analyzing feature trajectories differs dramatically from these previous methods in that we use dense clouds of KLT [13] feature tracks. In this way our underlying feature representation is also much closer to Johansson's point lights. Sun et al. [22] recently used similar techniques to model SIFT-feature trajectories. However, they find a fixed-dimensional velocity description using the stationary distribution of a Markov chain velocity model. The stationary distribution is closer to a velocity histogram than the more temporally structured representation we propose in Section 3.

3. Velocity Histories of Tracked Keypoints

We present a system that tracks a set of keypoints in a video sequence, extracts their velocity histories, and uses a generative mixture model to learn a velocity-history language and classify video sequences.

3.1. Feature Detection and Tracking

Given a video sequence, we extract feature trajectories using Birchfield's implementation [3] of the KLT tracker [13]. This system finds interest points where both eigenvalues of the matrix

$$\nabla I = \begin{pmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{pmatrix}$$

are greater than a threshold, and tracks them by calculating frame-to-frame translation with an affine consistency check. In order to maximize the duration of our feature trajectories, and thus capitalize on their descriptive power, we did not use the affine consistency check (which prevents feature deformation). This made feature trajectories noisier, but allowed them to capture non-rigid motion. While the detector finds both moving and nonmoving corners, our mixture model ensures that nonmoving features are described by a small number of mixture components. As described by Baker and Matthews [1], the tracker finds the match that minimizes

$$\sum_x [T(x) - I(W(x;p))]^2 \tag{1}$$

where T is the appearance of the feature to be matched in the last frame, I is the current frame, x is the position in the template window, W are the set of transformations considered (in this case, translation) between the last frame and the current one, and p are the parameters for the transformation. To derive the tracking update equations, we first express Equation 1 as an iterative update problem (replacing p with $p + \Delta p$), and Taylor expand the iterative update:

$$\sum_x [T(x) - I(w(x; p + \Delta p))]^2 \approx \sum_x [T(x) - I(w(x; p)) - \nabla I \frac{\partial W}{\partial p} \Delta p]^2 \quad (2)$$

Now we take the derivative of Equation 2 with respect to the update parameters:

$$\frac{\partial}{\partial p} \sum_x [T(x) - I(w(x; p)) - \nabla I \frac{\partial W}{\partial p} \Delta p]^2 = \sum_x \left[\nabla I \frac{\partial W}{\partial p}\right]^T \left[T(x) - I(w(x; p)) - \nabla I \frac{\partial W}{\partial p} \Delta p\right] \quad (3)$$

Setting the derivative in Equation 3 to zero and solving for the parameter update ΔP gives us:

$$\Delta p = \left[\sum_x SD^T(x) SD(x)\right]^{-1} \cdot \sum_x SD^T(x) T(x) - I(W(x; p)) \quad (4)$$

where $SD(x) = \nabla I \frac{\partial W}{\partial p}$ is the steepest descents image[1]. By iteratively updating the parameters according to Equation 4, we can quickly find the best matching for a tracked feature. Recent GPU-based implementations of this algorithm can run in real time [21].

We tracked 500 features at a time, detecting new features to replace lost tracks. We call each feature's quantized velocity over time its "velocity history", and use this as our basic feature. We emphasize that we are unaware of other work on activity recognition using a motion description with as long a temporal range as the technique presented here. Uniform quantization is done in log-polar coordinates, with 8 bins for direction, and 5 for magnitude. The temporal resolution of input video was maintained. We limited the velocity magnitude to 10 pixels per frame, which in all of our datasets corresponds almost exclusively to noise. This limit meant that for an individual feature's motion from one frame to the next, any motion above the threshold was not ignored, but that feature's motion on that frame was placed in the bin for the greatest motion magnitude in the appropriate direction. We also only considered feature trajectories lasting at least ten frames. This heuristic reduces noise while eliminating little of the extended velocity information we seek to capture. Examples of the trajectories being extracted can be seen in Figure 7.

To give an idea of the long temporal range of our feature flows, in the new dataset we introduce, containing videos between 10 and 60 seconds (300 and 1800 frames) long, the total number of flows extracted per activity sequence varied between about 700 and 2500, with an average of over 1400 flows per sequence. The mean duration of the flows was over 150 frames.

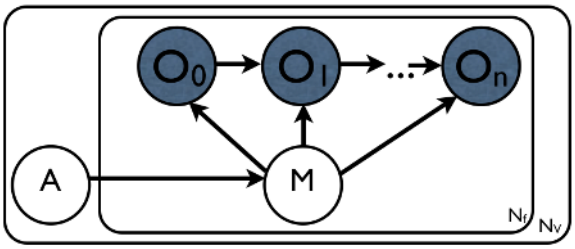

Figure 1. Graphical model for our tracked keypoint velocity history model (Dark circles denote observed variables).

3.2. Activity recognition

We model activity classes as a weighted mixture of bags of augmented trajectory sequences. Each activity model has a distribution over 100 shared mixture components. These mixture components can be thought of as velocity history "words", with each velocity history feature being generated by one mixture component, and each activity class having a distribution over these mixture components. Each mixture component models a velocity history feature with a Markov chain. So each velocity history feature is explained by a mixture of velocity history markov chains, and each activity class has a distribution over those mixture components. Our model is similar to a supervised latent Dirichlet allocation [5], where each activity is a document class with a distribution over mixture components (analogous to topics). Each mixture component has a distribution over velocity history features (analogous to words). In this way, each tracked velocity history feature has a different probability of being generated by each mixture component. Each document is modeled using naive Bayes as a "bag" of these soft-assigned mixture-component words. This model assumes that each document class shares a single multinomial distribution over mixture components.

3.2.1 Inference

Our model for action class is shown in Figure 1. Each action class is a mixture model. Each instance of an action (video clip) is treated as a bag of velocity history sequences. A is the action variable, which can take a value for each distinct action the system recognize. M is the mixture variable, indicating one of the action's mixture components. We use the notation M_f^i to abbreviate $M_f = i$, or in words, feature f is generated by mixture component i. O_n denotes a feature's discretized velocity at time n. N_f denotes the number of augmented features in the video sequence, and N_v denotes the number of video sequences.

Our joint probability model for an action is:

$$P(A,O) = \sum_M P(A,M,O) = P(A)\prod_f^{N_f}\sum_i^{N_m} P(M_f^i|A)P(O_{0,f}|M_f^i) \prod_{t=1}^{T_f} P(O_{t,f}|O_{t-1,f},M_f^i) \quad (5)$$

where N_m is the number of mixture components, N_f is the number of features, and T_f is the number of observations in feature trajectory f. For a feature tracked over 20 frames, t would have 19 disctinct values, since each observation is given by the difference between the feature's location between two frames. Note that all random variables are discrete, and all component distributions in the joint distribution are multinomial.

We train the model using Expectation Maximization (EM) [2], and classify a novel video sequence D by finding the action A that maximizes $P(A|D) \propto P(D|A)P(A)$. We assume a uniform prior on action, so $\text{argmax}_A P(A|D) \propto \text{argmax}_A P(D|A)$.

We also give Dirichlet priors to each of the model parameters in order to smooth over the sparse data. These priors are equivalent to having seen each next velocity state (within the same mixture component) once for each velocity state in each mixture component, and each mixture component once in each activity class.

While our velocity history features may seem to be very sparse compared to spatio-temporal patch features, their long duration means that they can contain a lot of information. Additionally, because of their simplicity, they could potentially be implemented very efficiently.

4. Feature Evaluation

We evaluated the performance of our model on the KTH dataset, a popular dataset introduced by Schuldt *et al*. [20]. This dataset consists of low-resolution (160 $\times$ 120 pixels) monochrome video clips taken by a fixed camera of 25 subjects performing each of 6 actions (walking, clapping, boxing, waving, jogging and running) in several size and background conditions. While our velocity history features are intended to perform well on high resolution video of complicated actions, it is important to evaluate the baseline performance of our feature on an established dataset. We use one of the popular testing paradigms for this dataset, where a system is trained on 8 randomly selected subjects, and tested on 9 randomly selected remaining subjects.

For comparison, we use Dollar *et al*.'s spatio-temporal cuboids [7], and Laptev *et al*.'s space-time interest points [12]. In order to ensure that the comparison is of the features, rather than the model, we use the same codebook size, 100 elements, and the same higher level model, naive bayes, for all systems.

Table 1. Performance results of different features using a train on 8, test on 9 testing paradigm on the KTH dataset. All features used 100 codewords and were combined using naive Bayes.

Method	Percent Correct
Spatio-Temporal Cuboids [7]	66
Velocity Histories (Sec. 3)	74
Space-Time Interest Points [12]	80

Results are shown in table 1. Clearly, our velocity history features are competitive with state-of-the-art local features on established datasets. We note that both Dollar *et al*.'s cuboids and Laptev *et al*.'s interest points have previously achieved much greater performance on this dataset using larger codebooks and more sophisticated discriminative models (like SVMs). We held the codebook size and feature combination model fixed in order to evaluate the features themselves.

5. High Resolution Videos of Complex Activities

Existing activity recognition datasets involve low-resolution videos of relatively simple actions. The KTH dataset [20], probably the single most popular one, is a case in point. This dataset consists of low-resolution (160 $\times$ 120 pixels) video clips taken by a non-moving camera at 25 frames per second of 25 subjects performing each of 6 actions (walking, clapping, boxing, waving, jogging and running) in several conditions varying scale, background, and clothing. Subjects repeated the actions several times, in clips ranging from just under 10 seconds, to just under a minute. The KTH dataset's relatively low resolution rewards techniques that focus on coarse, rather than fine spatial information. It doesn't include object interaction, an important part of many real-world activities. Lastly, its simple activities, while still a challenge to existing systems, may not reflect the complexity of the real activities a system might want to recognize.

We contribute a high resolution video dataset of activities of daily living[1]. These activities were selected to be useful for an assisted cognition task, and to be difficult to separate on the basis of any single source of information. For example, motion information might distinguish eating a banana from peeling a banana, but might have difficulty distinguishing eating a banana from eating snack chips, while appearance information would conversely be more useful in the latter than the former task.

[1] The dataset is available at http://www.cs.rochester.edu/∼rmessing/uradl

The full list of activities is: answering a phone, dialing a phone, looking up a phone number in a telephone directory, writing a phone number on a whiteboard, drinking a glass of water, eating snack chips, peeling a banana, eating a banana, chopping a banana, and eating food with silverware.

These activities were each performed three times by five different people. These people were all members a department of computer science, and were naive to the details of our model when the data was collected. By using people of different shapes, sizes, genders, and ethnicities, we hoped to ensure sufficient appearance variation to make the individual appearance features generated by each person different. We also hoped to ensure that activity models were robust to individual variation.

The scene was shot from about two meters away by a tripod-mounted camera. Video was taken at 1280 $\times$ 720 pixel resolution, at 30 frames per second. An example background image is shown in Figure 3.

The total number of features extracted per activity sequence varied between about 700 and 2500, averaging over 1400 features per sequence. The mean duration of the trajectories exceeded 150 frames. Video sequences lasted between 10 and 60 seconds, ending when the activity was completed.

Our evaluation consisted of training on all repetitions of activities by four of the five subjects, and testing on all repetitions of the fifth subject's activities. This leave-one-out testing was averaged over the performance with each left-out subject.

In addition to our tracked keypoint velocity history features, and the local spatio-temporal patch features of Dollar *et al*. [7] and Laptev *et al*. [12], we use Bobick and Davis's temporal templates [6], an approach based on the shape of tables of the presence or recency of motion at each location. Temporal templates, which offer a single-feature description of a video sequence, are not amenable to the same models as feature-based approaches. We used a standard technique with temporal templates, the Mahalanobis distance between the concatenated Hu moment representations of the motion energy and motion history images of a test video and the members of each activity class.

Due to limitations of the distributed implementation of Laptev *et al*.'s space-time interest point features, we were forced to reduce the video resolution to 640 $\times$ 480 pixels. However, we found almost no performance difference running Dollar *et al*.'s spatio-temporal cuboids at both this reduced resolution, and the full video resolution, suggesting that Laptev *et al*.'s features would also perform similarly on the full resolution video. Additionally, we found almost no performance difference running Laptev *et al*.'s space-time interest point features at a greatly reduced video resolution (160 $\times$ 120 pixels, as in the KTH dataset), suggesting that the extra fine spatial detail provided by high resolution video is unhelpful for space-time interest points.

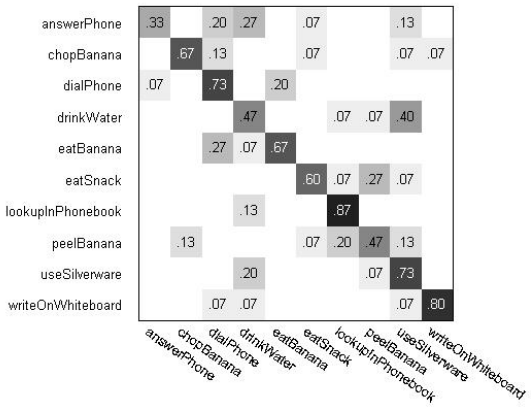

Figure 2. Confusion matrix using only feature trajectory information without augmentations. Overall accuracy is 63%. Zeros are omitted for clarity

Table 2. Performance results of different methods using a leave-one-out testing paradigm on our high resolution activities of daily living dataset

Method	Percent Correct
Temporal Templates [6]	33
Spatio-Temporal Cuboids [7]	36
Space-Time Interest Points [12]	59
Velocity Histories (Sec. 3)	63
Latent Velocity Histories (Sec. 7)	67
Augmented Velocity Histories (Sec. 6)	89

The performance of all systems on this dataset are shown in Table 2. A confusion matrix illustrating the performance of our velocity history features is shown in Figure 2. These results show several things. First, we see that feature-based methods, even when combined using the simplest models, outperform older whole-video methods. Given that the field has almost exclusively moved to feature-based methods, this is not surprising. Second, we see that while there is a range of performance among the local spatio-temporal features, velocity history features outperform them on this high resolution, high complexity dataset.

6. Augmenting our features with additional information

While the velocity history of tracked keypoints seems to be an effective feature by itself, it lacks any information about absolute position, appearance, color, or the position of important, nonmoving objects. By augmenting each of

Figure 3. A background image from the activities of daily living data set

Figure 4. Codebook-generated pixel map of birth-death locations

our velocity history features with this information, we can significantly improve performance.

6.1. Feature Augmentations

Features were augmented by location information indicating their initial and final positions (their "birth" and "death" positions). To obtain a given feature trajectory's birth and death, we use k-means clustering ($k = 24$) to form a codebook for both birth and death (independently), and assign a feature's birth and death to be it's nearest neighbor in the codebook. An example of this codebook can be seen in Figure 4, which was generated from activities in the scene shown in Figure 3.

In addition to this absolute position information, features are also augmented by relative position information. We use birth and death positions of the feature relative to the position of an unambiguously detected face, if present. Faces were detected by the OpenCV implementation of the Viola-Jones face detector [23], run independently on each frame of every video. Like birth and death, these positions are clustered by k-means ($k = 100$) into a codebook, and a feature position relative to the face is assigned to it's nearest neighbor in the codebook.

We augment features with local appearance information. We extract a 21×21 pixel patch around the initial position of the feature, normalized to the dominant orientation of the patch, calculate the horizontal and vertical gradients of the oriented patch, and use PCA-SIFT [11] to obtain a 40-dimensional descriptor. These descriptors are clustered by k-means into a codebook ($k = 400$), and each feature's patch is assigned to it's nearest neighbor in the codebook.

Lastly, we augment features with color information. From the same 21×21 pixel patch that we use for the appearance augmentation, we compute a color histogram in CIELAB space, with 10 bins in each color dimension. We use PCA to reduce the dimensionality of the histograms to 40 dimensions, and as with appearance, cluster these histograms using k-means into a 400 element codebook. Each feature flow patch's color histogram is assigned to its nearest neighbor in the codebook.

The codebooks for all augmentations were formed only using samples from the training data.

We note that only our absolute birth and death position

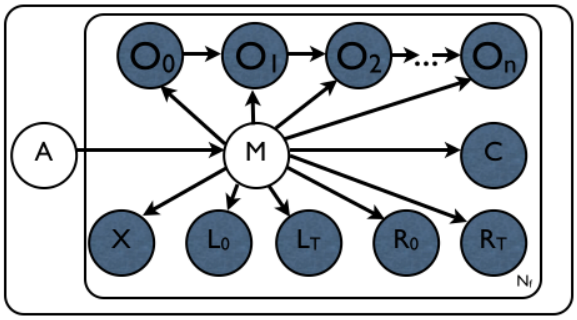

Figure 5. Graphical model for our augmented tracked keypoint velocity history model (Dark circles denote observed variables).

augmentations lack a significant degree of view invariance. The other augmentations, and the feature trajectory model itself, should maintain the same view invariance as other feature-based approaches to activity recognition which usually lack the scale and orientation invariance that characterize successful object recognition features.

We emphasize that our velocity history features are particularly appropriate recipients of these augmentations. By contrast, it is much less straightforward to augment local space-time features with these kinds of information, particularly because notions like "birth" and "death" position are not as well-defined for features based on local spatio-temporal patches.

6.2. Modeling Augmented Features

Our graphical model for our augmented velocity history model is shown in Figure 5. All symbols have the same meaning as in Figure 1, with the addition of L_0 and L_T as the birth and death location variables respectively, R_0 and R_T as the positions relative to the face at birth and death, X as the feature's appearance, and C as the feature's color. Augmentation information is combined using a naive Bayes assumption at the mixture component level, so the additional computational cost of each augmentation during inference is just another table lookup per feature.

Our new joint probability model for an action is:

$$
\begin{aligned}
&P(A, L_0, L_T, R_0, R_T, X, O) = \\
&\quad \sum_M P(A, M, L_0, L_T, R_0, R_T, X, O) = \\
&\quad P(A) \prod_f^{N_f} \sum_i^{N_m} P(M_f^i|A) P(L_{0,f}|M_f^i) P(L_{T,f}|M_f^i) \\
&\quad P(R_{0,f}|M_f^i) P(R_{T,f}|M_f^i) P(X_f|M_f^i) P(C_f|M_f^i) \\
&\quad P(O_{0,f}|M_f^i) \prod_{t=1}^{T} P(O_{t,f}|O_{t-1,f}, M_f^i) \qquad (6)
\end{aligned}
$$

As before, all random variables are discrete, and all component distributions in the joint distribution are multinomial.

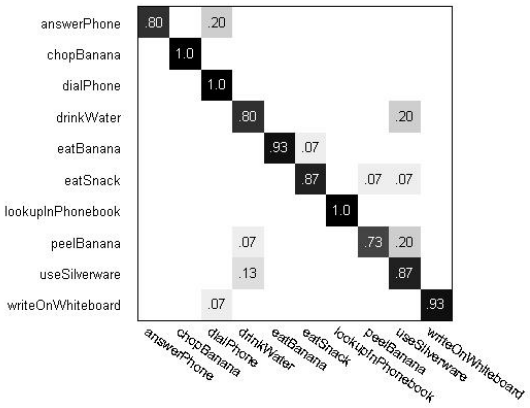

Figure 6. Confusion matrix for the augmented feature trajectory activity recognition system. Overall accuracy is 89%. Zeros are omitted for clarity

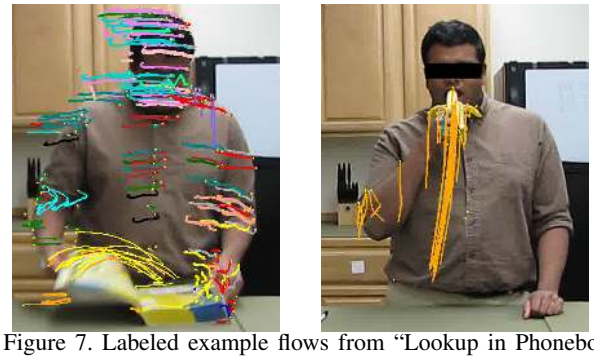

Figure 7. Labeled example flows from "Lookup in Phonebook" and "Eat Banana", with different colors denoting different mixture components. Note that complex semantic features like feature flow position relative to face are used in our framework.

6.3. Results

The performance of this augmented model using experimental conditions identical to those used for the unaugmented model is shown in Figure 6. Augmentations yield significant performance improvements on our high resolution activities of daily living dataset.

7. Adding uncertainty to the velocity history

While tracked keypoint velocity history has proven a powerful feature for activity recognition, particularly when augmented as in Section 6, it is dependent on the presence of sufficient data to learn a rich velocity transition matrix for all mixture components. One way to limit this dependence is to introduce uncertainty into each velocity observation. By modeling observed velocity as a true latent velocity plus fixed Gaussian noise, we can spread the probability mass from each observation over nearby, similar observations. As can be seen in Figure 8, this transforms our mixture of velocity Markov chains into a mixture of velocity hidden Markov models, or a factorial hidden Markov model.

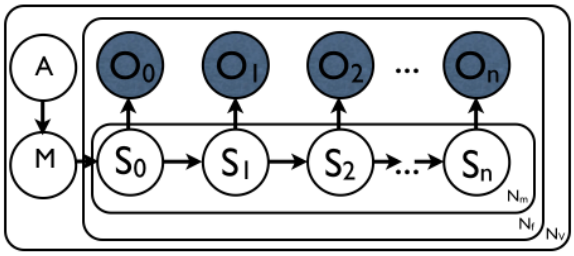

Figure 8. Graphical model for our latent tracked keypoint velocity history model (Dark circles denote observed variables).

7.1. Modeling Latent Veloicty

Our graphical model for our latent velocity history model is shown in Figure 8. All symbols have the same meaning as in Figure 1, and we introduce the variable S_t^i to indicate the latent velocity at time t as explained by mixture component i. By breaking up the latent velocities by mixture component, we avoid a cyclic dependence of each latent velocity state on the others, and so can use the forward-backward algorithm to compute latent state marginal distributions.

The joint PDF for an action with these new latent velocity states is:

$$P(A,S,O) = \sum_M P(A,M,S,O) = P(A)\prod_f^{N_f}\sum_i^{N_m} P(M_f^i|A)P(S_{0,f}^i|M_f^i)P(O_{0,f}|S_{0,f}^i) \prod_{t=1}^{T} P(S_{t,f}^i|S_{t-1,f}^i, M_f^i)P(O_{t,f}|S_{t,f}^i) \quad (7)$$

As in the previous models, all variables are discrete, and all distributions are multinomial, aside from the $P(O|S)$ term, which is the sum of two fixed zero-mean Gaussian distributions. One Gaussian is broad, to approximate uniform noise, with a diagonal covariance matrix whose entries are 10 pixels per frame, and the other is narrow, with entries one quarter the value of the broad one.

While this model is more powerful than the mixture of velocity Markov chains, it is much more expensive to perform inference in it. Each velocity transition table lookup in the mixture of velocity Markov chains becomes a matrix multiplication. In order to test this model, it was necessary to break apart the mixture components during training. Instead of 100 mixture components shared over all action classes, 10 mixture components were assigned to each action class, and trained only on data from that class. Testing used a naive Bayes model sharing mixture components across all action classes, as in the previous versions of the model.

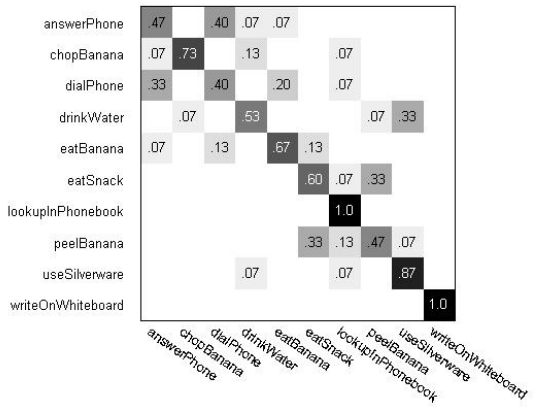

Figure 9. Confusion matrix for the latent feature trajectory activity recognition system. Overall accuracy is 67%. Zeros are omitted for clarity

7.2. Results

The performance of the latent velocity model is shown in Figure 9. Clearly, there is some improvement in performance over the model without latent states shown in Figure 2. This suggests that incorporating uncertainty in our velocity states, can offer moderately improved results. On the one hand, it is encouraging to see that even without adding more data, a more sophisticated model has the potential to improve performance. On the other hand, it is a modest improvement compared to that offered by the augmentations of Section 6, at a much greater computational cost.

8. Conclusions

The velocity history of tracked keypoints is clearly a useful feature for activity recognition particularly when augmented with extra information. By incorporating information outside a local spatio-temporal patch at the feature level, it allows a bag-of-features model to a significant amount of non-local structure. We anticipate that high resolution video data will become more commonplace, and that practical activity recognition systems will face more complicated activities. As the complixity of activity recognition increases, features capable of capturing non-local structure may prove more valuable than local spatio-temporal methods.

References

[1] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *IJCV*, 56(3):221 – 255, March 2004.

[2] J. Bilmes. A gentle tutorial on the EM algorithm. Technical Report ICSI-TR-97-021, University of Berkeley, 1997.

[3] S. Birchfield. Klt: An implementation of the kanade-lucas-tomasi feature tracker, 1996.

[4] M. Black and Y. Yacoob. Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. In *ICCV*, pages 374–381, 1995.

[5] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[6] A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *IEEE PAMI*, 23:257–267, 2001.

[7] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *VS-PETS*, October 2005.

[8] H. Jiang and D. R. Martin. Finding actions using shape flows. In *ECCV*, pages 278–292, 2008.

[9] G. Johansson. Visual perception of biological motion and a model for its analysis. *Perception and Psychophysics*, 14:201–211, 1973.

[10] I. Junejo, E. Dexter, I. Laptev, and P. Pérez. Cross-view action recognition from temporal self-similarities. In *ECCV*, volume 2, pages 293–306, Marseille, France, 2008.

[11] Y. Ke and R. Sukthankar. Pca-sift: a more distinctive representation for local image descriptors. In *CVPR*, pages II: 506–513, 2004.

[12] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *CVPR*, pages 1–8, Anchorage, Alaska, June 2008.

[13] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI*, pages 674–679, 1981.

[14] A. Madabhushi and J. K. Aggarwal. A bayesian approach to human activity recognition. In *VS '99: Workshop on Visual Surveillance*, page 25, 1999.

[15] J. C. Niebles and L. Fei-Fei. A hierarchical model model of shape and appearance for human action classification. In *CVPR*, 2007.

[16] R. Polana and R. C. Nelson. Detecting activities. In *CVPR*, 1993.

[17] D. Ramanan and D. A. Forsyth. Automatic annotation of everyday movements. In *NIPS*, December 2003.

[18] R. Rosales and S. Sclaroff. Trajectory guided tracking and recognition of actions. Technical report, Boston University, Boston, MA, USA, 1999.

[19] S. Savarese, A. D. Pozo, J. C. Niebles, and L. Fei-Fei. Spatial-temporal correlations for unsupervised action classification. In *Motion and Video Computing*, 2008.

[20] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local svm approach. In *ICPR*, pages 32–36, 2004.

[21] S. N. Sinha, J. michael Frahm, M. Pollefeys, and Y. Genc. Gpu-based video feature tracking and matching. Technical report, In Workshop on Edge Computing, 2006.

[22] J. Sun, X. Wu, S. Yan, L.-F. Cheong, T.-S. Chua, and J. Li. Hierarchical spatio-temporal context modeling for action recognition. In *CVPR*, 2009.

[23] P. Viola and M. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, May 2004.